Complexity Measurement

Type : java

File Name : code.java

File Path : /Users/Shark/Desktop/CodeCover/code.java

#	Line	D. Method	C. Method	Rec	T. M. C	Cs Tokens	Cs	Ctc	Cnc	Ci	TW	Cps	Cr	Cps Cr
1	public class Mountain extends Fractal {			false	0		0	0	0	0	0	0	0	0
2	private Point p1;			false	0	1,	1	0	0	3	3	3	0	3
3	private Point p2;			false	0	2,	1	0	0	3	3	3	0	3
4	private Point p3;			false	0	3,	1	0	0	3	3	3	0	3
5	private int dev;			false	0	int, dev,	2	0	0	3	3	6	0	6
6	private List<Side> sides;			false	0	> ,	1	0	0	3	3	3	0	3
7	private int level;			false	0	int, level,	2	0	0	3	3	6	0	6
8				false	0		0	0	0	0	0	0	0	0
9	public Mountain(int dev, Point p1, Point p2, Point p3) {	Mountain		false	0	1, 2, 3, int, dev,	5	0	0	3	3	15	0	15
10	super();		super	false	0		0	0	0	3	3	0	0	0
11	this.dev = dev;			false	0	=, .d, dev, dev,	4	0	0	3	3	12	0	12
12	this.p1 = p1;			false	0	1, 1, =, .p,	4	0	0	3	3	12	0	12
13	this.p2 = p2;			false	0	2, 2, =, .p,	4	0	0	3	3	12	0	12
14	this.p3 = p3;			false	0	3, 3, =, .p,	4	0	0	3	3	12	0	12
15	sides = new LinkedList<Side>();			false	0	=, >(, new,	3	0	0	3	3	9	0	9
16	level = 0;			false	0	0, =, level,	3	0	0	3	3	9	0	9
17	}			false	0		0	0	0	0	0	0	0	0
18				false	0		0	0	0	0	0	0	0	0
19	@Override			false	0		0	0	0	3	3	0	0	0
20	public String getTitle() {	getTitle		false	0	String, getTitle,	2	0	0	3	3	6	0	6
21	return "Snett berg";			false	0	"Snett berg",	1	0	0	3	3	3	0	3
22	}			false	0		0	0	0	0	0	0	0	0
23	private void fractalTriangle(TurtleGraphics turtle, int order, int dev, Point p1, Point p2, Point p3, int level) {	fractalTriangle		true	12	1, 2, 3, void, int, int, int, dev, level, fractalTriangle, order,	11	0	0	3	3	33	792	825
24	if (order == 0) {			false	0	0, ==, if, order,	4	1	1	3	5	20	0	20
25	turtle.moveTo(p1.getX(), p1.getY());		moveTo	false	0	1, 1, .m, .g, .g,	5	0	1	3	4	20	0	20
26	turtle.penDown();		penDown	false	0	.p,	1	0	1	3	4	4	0	4
27	turtle.forwardTo(p2.getX(), p2.getY());		forwardTo	false	0	2, 2, .f, .g, .g, for,	6	0	1	3	4	24	0	24
28	turtle.forwardTo(p3.getX(), p3.getY());		forwardTo	false	0	3, 3, .f, .g,	6	0	1	3	4	24	0	24

#	Line	D. Method	C. Method	Rec	T. M. C	Cs Tokens	Cs	Ctc	Cnc	Ci	TW	Cps	Cr	Cps Cr
						.g, for,								
29	turtle.forwardTo(p1.getX(), p1.getY());		forwardTo	false	0	1, 1, .f, .g, .g, for,	6	0	1	3	4	24	0	24
30	turtle.penUp();		penUp	false	0	.p,	1	0	1	3	4	4	0	4
31	setLevel(level);		setLevel	false	0	level,	1	0	1	3	4	4	0	4
32	System.out.println(sides.size());		println	false	0	.o, .p, .s, println, System, out,	6	0	1	3	4	24	0	24
33	} else {			false	0		0	0	1	0	1	0	0	0
34	double offset1 = RandomUtilities.randFunc(dev);		randFunc	false	0	1, =, .r, double, dev, offset,	6	0	1	3	4	24	0	24
35	double offset2 = RandomUtilities.randFunc(dev);		randFunc	false	0	2, =, .r, double, dev, offset,	6	0	1	3	4	24	0	24
36	double offset3 = RandomUtilities.randFunc(dev);		randFunc	false	0	3, =, .r, double, dev, offset,	6	0	1	3	4	24	0	24
37	dev /= 2;			false	0	2, /=, dev,	3	0	1	3	4	12	0	12
38	Point mLeft;			false	0		0	0	1	3	4	0	0	0
39	Point mRight;			false	0		0	0	1	3	4	0	0	0
40	Point mBottom;			false	0		0	0	1	3	4	0	0	0
41				false	0		0	0	1	0	1	0	0	0
42	mLeft = getMPoint(p1, p2, offset1, level);		getMPoint	false	0	1, 2, 1, =, level, offset,	6	0	1	3	4	24	0	24
43	mRight = getMPoint(p2, p3, offset2, level);		getMPoint	false	0	2, 3, 2, =, level, offset,	6	0	1	3	4	24	0	24
44	mBottom = getMPoint(p1, p3, offset3, level);		getMPoint	false	0	1, 3, 3, =, level, offset,	6	0	1	3	4	24	0	24
45	fractalTriangle(turtle, order - 1, dev, mLeft, p2, mRight, level + 1);		fractalTriangle	true	0	1, 2, 1, + , - , dev, level, fractalTriangle, order,	9	0	1	3	4	36	0	36
46	fractalTriangle(turtle, order - 1, dev, mLeft, mRight, mBottom, level + 1);		fractalTriangle	true	0	1, 1, + , - , dev, level, fractalTriangle, order,	8	0	1	3	4	32	0	32
47	fractalTriangle(turtle, order - 1, dev, p1, mLeft, mBottom, level + 1);		fractalTriangle	true	0	1, 1, 1, + , - , dev, level, fractalTriangle, order,	9	0	1	3	4	36	0	36
48	fractalTriangle(turtle, order - 1, dev, mBottom, mRight, p3, level + 1);		fractalTriangle	true	0	1, 3, 1, + , - , dev, level, fractalTriangle, order,	9	0	1	3	4	36	0	36
49	}			false	0		0	0	1	0	1	0	0	0
50	mLeft = getMPoint(p1, p2, offset1, level);		getMPoint	false	0	1, 2, 1, =, level,	6	0	0	3	3	18	0	18

#	Line	D. Method	C. Method	Rec	T. M. C	Cs Tokens	Cs	Ctc	Cnc	Ci	TW	Cps	Cr	Cps Cr
						offset,								
51	mRight = getMPoint(p2, p3, offset2, level);		getMPoint	false	0	2, 3, 2, =, level, offset,	6	0	0	3	3	18	0	18
52	}			false	0		0	0	0	0	0	0	0	0
53				false	0		0	0	0	0	0	0	0	0
54	private Point getMPoint(Point p1, Point p2, double offset, int level){	getMPoint		true	18	1, 2, double, int, level, offset,	6	0	0	3	3	18	648	666
55	Side test = new Side(p1, p2);		Side	false	0	1, 2, =, new,	4	0	0	3	3	12	0	12
56	int index = sides.indexOf(test);		indexOf	false	0	=, .i, int, index, index,	5	0	0	3	3	15	0	15
57	if (index >= 0) {			false	0	0, >=, if, index,	4	1	1	3	5	20	0	20
58	Side temp = sides.get(index);		get	false	0	=, .g, index,	3	0	1	3	4	12	0	12
59	sides.remove(index);		remove	false	0	.r, index,	2	0	1	3	4	8	0	8
60	return temp.getMPoint();		getMPoint	true	0	.g,	1	0	1	3	4	4	0	4
61	} else {			false	0		0	0	1	0	1	0	0	0
62	Point temp = new Point((p1.getX() + p2.getX()) / 2.0,		Point	false	0	1, 2, 2, 0, + , / , =, .g, .g, .0, new,	11	0	1	3	4	44	0	44
63	((p1.getY() + p2.getY()) / 2.0) + offset);		getY	false	0	1, 2, 2, 0, + , + , / , .g, .g, .0, offset,	11	0	1	3	4	44	0	44
64	sides.add(new Side(p1, p2, temp, level));		add	false	0	1, 2, .a, new, level,	5	0	1	3	4	20	0	20
65	return temp;			false	0		0	0	1	3	4	0	0	0
66	}			false	0		0	0	1	0	1	0	0	0
67	sides.remove(index);		remove	false	0	.r, index,	2	0	0	3	3	6	0	6
68	}			false	0		0	0	0	0	0	0	0	0
69				false	0		0	0	0	0	0	0	0	0
70	private void setLevel(int level){	setLevel		false	3	void, int, level, setLevel,	4	0	0	3	3	12	0	12
71	Iterator<Side> itr = sides.iterator();		iterator	false	0	=, .i, > ,	3	0	0	3	3	9	0	9
72	while(itr.hasNext()){		hasNext	false	0	.h, while,	2	2	1	3	6	12	0	12
73	Side temp = itr.next();		next	false	0	=, .n,	2	0	1	3	4	8	0	8
74	if(temp.getLevel() >= level){		getLevel	false	0	.g, >=, if, level,	4	1	2	3	6	24	0	24
75	itr.remove();		remove	false	0	.r,	1	0	2	3	5	5	0	5
76	}			false	0		0	0	2	0	2	0	0	0
77	}			false	0		0	0	1	0	1	0	0	0
78	}			false	0		0	0	0	0	0	0	0	0
79	}			false	0		0	0	0	0	0	0	0	0
						Total CP								2340

Type : java

File Name : code.java

File Path : /Users/Shark/Desktop/CodeCover/Templates/code.java

---

#	Line	D. Method	C. Method	Rec	T. M. C	Cs Tokens	Cs	Ctc	Cnc	Ci	TW	Cps	Cr	Cps Cr
1	public class Mountain extends Fractal {			false	0		0	0	0	0	0	0	0	0
2	private Point p1;			false	0	1,	1	0	0	3	3	3	0	3
3	private Point p2;			false	0	2,	1	0	0	3	3	3	0	3
4	private Point p3;			false	0	3,	1	0	0	3	3	3	0	3
5	private int dev;			false	0	int, dev,	2	0	0	3	3	6	0	6
6	private List<Side> sides;			false	0	> ,	1	0	0	3	3	3	0	3
7	private int level;			false	0	int, level,	2	0	0	3	3	6	0	6
8				false	0		0	0	0	0	0	0	0	0
9	public Mountain(int dev, Point p1, Point p2, Point p3) {	Mountain		false	0	1, 2, 3, int, dev,	5	0	0	3	3	15	0	15
10	super();		super	false	0		0	0	0	3	3	0	0	0
11	this.dev = dev;			false	0	=, .d, dev, dev,	4	0	0	3	3	12	0	12
12	this.p1 = p1;			false	0	1, 1, =, .p,	4	0	0	3	3	12	0	12
13	this.p2 = p2;			false	0	2, 2, =, .p,	4	0	0	3	3	12	0	12
14	this.p3 = p3;			false	0	3, 3, =, .p,	4	0	0	3	3	12	0	12
15	sides = new LinkedList<Side>();			false	0	=, >(, new,	3	0	0	3	3	9	0	9
16	level = 0;			false	0	0, =, level,	3	0	0	3	3	9	0	9
17	}			false	0		0	0	0	0	0	0	0	0
18				false	0		0	0	0	0	0	0	0	0
19	@Override			false	0		0	0	0	3	3	0	0	0
20	public String getTitle() {	getTitle		false	0	String, getTitle,	2	0	0	3	3	6	0	6
21	return "Snett berg";			false	0	"Snett berg",	1	0	0	3	3	3	0	3
22	}			false	0		0	0	0	0	0	0	0	0
23	private void fractalTriangle(TurtleGraphics turtle, int order, int dev, Point p1, Point p2, Point p3, int level) {	fractalTriangle		true	12	1, 2, 3, void, int, int, int, dev, level, fractalTriangle, order,	11	0	0	3	3	33	792	825
24	if (order == 0) {			false	0	0, ==, if, order,	4	1	1	3	5	20	0	20
25	turtle.moveTo(p1.getX(), p1.getY());		moveTo	false	0	1, 1, .m, .g, .g,	5	0	1	3	4	20	0	20
26	turtle.penDown();		penDown	false	0	.p,	1	0	1	3	4	4	0	4
27	turtle.forwardTo(p2.getX(), p2.getY());		forwardTo	false	0	2, 2, .f, .g, .g, for,	6	0	1	3	4	24	0	24
28	turtle.forwardTo(p3.getX(), p3.getY());		forwardTo	false	0	3, 3, .f, .g, .g, for,	6	0	1	3	4	24	0	24
29	turtle.forwardTo(p1.getX(), p1.getY());		forwardTo	false	0	1, 1, .f, .g, .g, for,	6	0	1	3	4	24	0	24

#	Line	D. Method	C. Method	Rec	T. M. C	Cs Tokens	Cs	Ctc	Cnc	Ci	TW	Cps	Cr	Cps Cr
30	turtle.penUp();		penUp	false	0	.p,	1	0	1	3	4	4	0	4
31	setLevel(level);		setLevel	false	0	level,	1	0	1	3	4	4	0	4
32	System.out.println(sides.size());		println	false	0	.o, .p, .s, println, System, out,	6	0	1	3	4	24	0	24
33	} else {			false	0		0	0	1	0	1	0	0	0
34	double offset1 = RandomUtilities.randFunc(dev);		randFunc	false	0	1, =, .r, double, dev, offset,	6	0	1	3	4	24	0	24
35	double offset2 = RandomUtilities.randFunc(dev);		randFunc	false	0	2, =, .r, double, dev, offset,	6	0	1	3	4	24	0	24
36	double offset3 = RandomUtilities.randFunc(dev);		randFunc	false	0	3, =, .r, double, dev, offset,	6	0	1	3	4	24	0	24
37	dev /= 2;			false	0	2, /=, dev,	3	0	1	3	4	12	0	12
38	Point mLeft;			false	0		0	0	1	3	4	0	0	0
39	Point mRight;			false	0		0	0	1	3	4	0	0	0
40	Point mBottom;			false	0		0	0	1	3	4	0	0	0
41				false	0		0	0	1	0	1	0	0	0
42	mLeft = getMPoint(p1, p2, offset1, level);		getMPoint	false	0	1, 2, 1, =, level, offset,	6	0	1	3	4	24	0	24
43	mRight = getMPoint(p2, p3, offset2, level);		getMPoint	false	0	2, 3, 2, =, level, offset,	6	0	1	3	4	24	0	24
44	mBottom = getMPoint(p1, p3, offset3, level);		getMPoint	false	0	1, 3, 3, =, level, offset,	6	0	1	3	4	24	0	24
45	fractalTriangle(turtle, order - 1, dev, mLeft, p2, mRight, level + 1);		fractalTriangle	true	0	1, 2, 1, + , - , dev, level, fractalTriangle, order,	9	0	1	3	4	36	0	36
46	fractalTriangle(turtle, order - 1, dev, mLeft, mRight, mBottom, level + 1);		fractalTriangle	true	0	1, 1, + , - , dev, level, fractalTriangle, order,	8	0	1	3	4	32	0	32
47	fractalTriangle(turtle, order - 1, dev, p1, mLeft, mBottom, level + 1);		fractalTriangle	true	0	1, 1, 1, + , - , dev, level, fractalTriangle, order,	9	0	1	3	4	36	0	36
48	fractalTriangle(turtle, order - 1, dev, mBottom, mRight, p3, level + 1);		fractalTriangle	true	0	1, 3, 1, + , - , dev, level, fractalTriangle, order,	9	0	1	3	4	36	0	36
49	}			false	0		0	0	1	0	1	0	0	0
50	mLeft = getMPoint(p1, p2, offset1, level);		getMPoint	false	0	1, 2, 1, =, level, offset,	6	0	0	3	3	18	0	18
51	mRight = getMPoint(p2, p3, offset2, level);		getMPoint	false	0	2, 3, 2, =, level,	6	0	0	3	3	18	0	18

#	Line	D. Method	C. Method	Rec	T. M. C	Cs Tokens	Cs	Ctc	Cnc	Ci	TW	Cps	Cr	Cps Cr
						offset,								
52	}			false	0		0	0	0	0	0	0	0	0
53				false	0		0	0	0	0	0	0	0	0
54	private Point getMPoint(Point p1, Point p2, double offset, int level){	getMPoint		true	18	1, 2, double, int, level, offset,	6	0	0	3	3	18	648	666
55	Side test = new Side(p1, p2);		Side	false	0	1, 2, =, new,	4	0	0	3	3	12	0	12
56	int index = sides.indexOf(test);		indexOf	false	0	=, .i, int, index, index,	5	0	0	3	3	15	0	15
57	if (index >= 0) {			false	0	0, >=, if, index,	4	1	1	3	5	20	0	20
58	Side temp = sides.get(index);		get	false	0	=, .g, index,	3	0	1	3	4	12	0	12
59	sides.remove(index);		remove	false	0	.r, index,	2	0	1	3	4	8	0	8
60	return temp.getMPoint();		getMPoint	true	0	.g,	1	0	1	3	4	4	0	4
61	} else {			false	0		0	0	1	0	1	0	0	0
62	Point temp = new Point((p1.getX() + p2.getX()) / 2.0,		Point	false	0	1, 2, 2, 0, + , / , =, .g, .g, .0, new,	11	0	1	3	4	44	0	44
63	((p1.getY() + p2.getY()) / 2.0) + offset);		getY	false	0	1, 2, 2, 0, + , + , / , .g, .g, .0, offset,	11	0	1	3	4	44	0	44
64	sides.add(new Side(p1, p2, temp, level));		add	false	0	1, 2, .a, new, level,	5	0	1	3	4	20	0	20
65	return temp;			false	0		0	0	1	3	4	0	0	0
66	}			false	0		0	0	1	0	1	0	0	0
67	sides.remove(index);		remove	false	0	.r, index,	2	0	0	3	3	6	0	6
68	}			false	0		0	0	0	0	0	0	0	0
69				false	0		0	0	0	0	0	0	0	0
70	private void setLevel(int level){	setLevel		false	3	void, int, level, setLevel,	4	0	0	3	3	12	0	12
71	Iterator<Side> itr = sides.iterator();		iterator	false	0	=, .i, > ,	3	0	0	3	3	9	0	9
72	while(itr.hasNext()){		hasNext	false	0	.h, while,	2	2	1	3	6	12	0	12
73	Side temp = itr.next();		next	false	0	=, .n,	2	0	1	3	4	8	0	8
74	if(temp.getLevel() >= level){		getLevel	false	0	.g, >=, if, level,	4	1	2	3	6	24	0	24
75	itr.remove();		remove	false	0	.r,	1	0	2	3	5	5	0	5
76	}			false	0		0	0	2	0	2	0	0	0
77	}			false	0		0	0	1	0	1	0	0	0
78	}			false	0		0	0	0	0	0	0	0	0
79	}			false	0		0	0	0	0	0	0	0	0
						Total CP								2340

Type : java

File Name : code.java

File Path : /Users/Shark/Desktop/CodeCover/Test Results/code.java

---

#	Line	D. Method	C. Method	Rec	T. M. C	Cs Tokens	Cs	Ctc	Cnc	Ci	TW	Cps	Cr	Cps Cr
1	public class Mountain extends Fractal {			false	0		0	0	0	0	0	0	0	0
2	private Point p1;			false	0	1,	1	0	0	3	3	3	0	3
3	private Point p2;			false	0	2,	1	0	0	3	3	3	0	3
4	private Point p3;			false	0	3,	1	0	0	3	3	3	0	3
5	private int dev;			false	0	int, dev,	2	0	0	3	3	6	0	6
6	private List<Side> sides;			false	0	> ,	1	0	0	3	3	3	0	3
7	private int level;			false	0	int, level,	2	0	0	3	3	6	0	6
8				false	0		0	0	0	0	0	0	0	0
9	public Mountain(int dev, Point p1, Point p2, Point p3) {	Mountain		false	0	1, 2, 3, int, dev,	5	0	0	3	3	15	0	15
10	super();		super	false	0		0	0	0	3	3	0	0	0
11	this.dev = dev;			false	0	=, .d, dev, dev,	4	0	0	3	3	12	0	12
12	this.p1 = p1;			false	0	1, 1, =, .p,	4	0	0	3	3	12	0	12
13	this.p2 = p2;			false	0	2, 2, =, .p,	4	0	0	3	3	12	0	12
14	this.p3 = p3;			false	0	3, 3, =, .p,	4	0	0	3	3	12	0	12
15	sides = new LinkedList<Side>();			false	0	=, >(, new,	3	0	0	3	3	9	0	9
16	level = 0;			false	0	0, =, level,	3	0	0	3	3	9	0	9
17	}			false	0		0	0	0	0	0	0	0	0
18				false	0		0	0	0	0	0	0	0	0
19	@Override			false	0		0	0	0	3	3	0	0	0
20	public String getTitle() {	getTitle		false	0	String, getTitle,	2	0	0	3	3	6	0	6
21	return "Snett berg";			false	0	"Snett berg",	1	0	0	3	3	3	0	3
22	}			false	0		0	0	0	0	0	0	0	0
23	private void fractalTriangle(TurtleGraphics turtle, int order, int dev, Point p1, Point p2, Point p3, int level) {	fractalTriangle		true	12	1, 2, 3, void, int, int, int, dev, level, fractalTriangle, order,	11	0	0	3	3	33	792	825
24	if (order == 0) {			false	0	0, ==, if, order,	4	1	1	3	5	20	0	20
25	turtle.moveTo(p1.getX(), p1.getY());		moveTo	false	0	1, 1, .m, .g, .g,	5	0	1	3	4	20	0	20
26	turtle.penDown();		penDown	false	0	.p,	1	0	1	3	4	4	0	4
27	turtle.forwardTo(p2.getX(), p2.getY());		forwardTo	false	0	2, 2, .f, .g, .g, for,	6	0	1	3	4	24	0	24
28	turtle.forwardTo(p3.getX(), p3.getY());		forwardTo	false	0	3, 3, .f, .g, .g, for,	6	0	1	3	4	24	0	24
29	turtle.forwardTo(p1.getX(), p1.getY());		forwardTo	false	0	1, 1, .f, .g, .g, for,	6	0	1	3	4	24	0	24

#	Line	D. Method	C. Method	Rec	T. M. C	Cs Tokens	Cs	Ctc	Cnc	Ci	TW	Cps	Cr	Cps Cr
30	turtle.penUp();		penUp	false	0	.p,	1	0	1	3	4	4	0	4
31	setLevel(level);		setLevel	false	0	level,	1	0	1	3	4	4	0	4
32	System.out.println(sides.size());		println	false	0	.o, .p, .s, println, System, out,	6	0	1	3	4	24	0	24
33	} else {			false	0		0	0	1	0	1	0	0	0
34	double offset1 = RandomUtilities.randFunc(dev);		randFunc	false	0	1, =, .r, double, dev, offset,	6	0	1	3	4	24	0	24
35	double offset2 = RandomUtilities.randFunc(dev);		randFunc	false	0	2, =, .r, double, dev, offset,	6	0	1	3	4	24	0	24
36	double offset3 = RandomUtilities.randFunc(dev);		randFunc	false	0	3, =, .r, double, dev, offset,	6	0	1	3	4	24	0	24
37	dev /= 2;			false	0	2, /=, dev,	3	0	1	3	4	12	0	12
38	Point mLeft;			false	0		0	0	1	3	4	0	0	0
39	Point mRight;			false	0		0	0	1	3	4	0	0	0
40	Point mBottom;			false	0		0	0	1	3	4	0	0	0
41				false	0		0	0	1	0	1	0	0	0
42	mLeft = getMPoint(p1, p2, offset1, level);		getMPoint	false	0	1, 2, 1, =, level, offset,	6	0	1	3	4	24	0	24
43	mRight = getMPoint(p2, p3, offset2, level);		getMPoint	false	0	2, 3, 2, =, level, offset,	6	0	1	3	4	24	0	24
44	mBottom = getMPoint(p1, p3, offset3, level);		getMPoint	false	0	1, 3, 3, =, level, offset,	6	0	1	3	4	24	0	24
45	fractalTriangle(turtle, order - 1, dev, mLeft, p2, mRight, level + 1);		fractalTriangle	true	0	1, 2, 1, + , - , dev, level, fractalTriangle, order,	9	0	1	3	4	36	0	36
46	fractalTriangle(turtle, order - 1, dev, mLeft, mRight, mBottom, level + 1);		fractalTriangle	true	0	1, 1, + , - , dev, level, fractalTriangle, order,	8	0	1	3	4	32	0	32
47	fractalTriangle(turtle, order - 1, dev, p1, mLeft, mBottom, level + 1);		fractalTriangle	true	0	1, 1, 1, + , - , dev, level, fractalTriangle, order,	9	0	1	3	4	36	0	36
48	fractalTriangle(turtle, order - 1, dev, mBottom, mRight, p3, level + 1);		fractalTriangle	true	0	1, 3, 1, + , - , dev, level, fractalTriangle, order,	9	0	1	3	4	36	0	36
49	}			false	0		0	0	1	0	1	0	0	0
50	mLeft = getMPoint(p1, p2, offset1, level);		getMPoint	false	0	1, 2, 1, =, level, offset,	6	0	0	3	3	18	0	18
51	mRight = getMPoint(p2, p3, offset2, level);		getMPoint	false	0	2, 3, 2, =, level,	6	0	0	3	3	18	0	18

#	Line	D. Method	C. Method	Rec	T. M. C	Cs Tokens	Cs	Ctc	Cnc	Ci	TW	Cps	Cr	Cps Cr
						offset,								
52	}			false	0		0	0	0	0	0	0	0	0
53				false	0		0	0	0	0	0	0	0	0
54	private Point getMPoint(Point p1, Point p2, double offset, int level){	getMPoint		true	18	1, 2, double, int, level, offset,	6	0	0	3	3	18	648	666
55	Side test = new Side(p1, p2);		Side	false	0	1, 2, =, new,	4	0	0	3	3	12	0	12
56	int index = sides.indexOf(test);		indexOf	false	0	=, .i, int, index, index,	5	0	0	3	3	15	0	15
57	if (index >= 0) {			false	0	0, >=, if, index,	4	1	1	3	5	20	0	20
58	Side temp = sides.get(index);		get	false	0	=, .g, index,	3	0	1	3	4	12	0	12
59	sides.remove(index);		remove	false	0	.r, index,	2	0	1	3	4	8	0	8
60	return temp.getMPoint();		getMPoint	true	0	.g,	1	0	1	3	4	4	0	4
61	} else {			false	0		0	0	1	0	1	0	0	0
62	Point temp = new Point((p1.getX() + p2.getX()) / 2.0,		Point	false	0	1, 2, 2, 0, + , / , =, .g, .g, .0, new,	11	0	1	3	4	44	0	44
63	((p1.getY() + p2.getY()) / 2.0) + offset);		getY	false	0	1, 2, 2, 0, + , + , / , .g, .g, .0, offset,	11	0	1	3	4	44	0	44
64	sides.add(new Side(p1, p2, temp, level));		add	false	0	1, 2, .a, new, level,	5	0	1	3	4	20	0	20
65	return temp;			false	0		0	0	1	3	4	0	0	0
66	}			false	0		0	0	1	0	1	0	0	0
67	sides.remove(index);		remove	false	0	.r, index,	2	0	0	3	3	6	0	6
68	}			false	0		0	0	0	0	0	0	0	0
69				false	0		0	0	0	0	0	0	0	0
70	private void setLevel(int level){	setLevel		false	3	void, int, level, setLevel,	4	0	0	3	3	12	0	12
71	Iterator<Side> itr = sides.iterator();		iterator	false	0	=, .i, > ,	3	0	0	3	3	9	0	9
72	while(itr.hasNext()){		hasNext	false	0	.h, while,	2	2	1	3	6	12	0	12
73	Side temp = itr.next();		next	false	0	=, .n,	2	0	1	3	4	8	0	8
74	if(temp.getLevel() >= level){		getLevel	false	0	.g, >=, if, level,	4	1	2	3	6	24	0	24
75	itr.remove();		remove	false	0	.r,	1	0	2	3	5	5	0	5
76	}			false	0		0	0	2	0	2	0	0	0
77	}			false	0		0	0	1	0	1	0	0	0
78	}			false	0		0	0	0	0	0	0	0	0
79	}			false	0		0	0	0	0	0	0	0	0
						Total CP								2340

Summary

Description	Count/Quantity
Total Number of files Read	3
Total Program Cp	7020

Acronyms

Acronym	Meaning
D. Method	Declared Method
C. Method	Calling Method
Rec	Is Recursive (True/False)
T.M.C	# Times Method was Called
Cs	Complexity of a program statement due to size
Ctc	Complexity of a program statement due to type of control structures
Cnc	Complexity of a program statement due to nesting of control structures
Ci	Complexity of a program statement due to inheritance
TW	Total complexity of a program statement
Cps	Complexity of a program statement
Cr	Complexity introduced due to recursion
CpsCr	Cps + Cr (Taken Line by line for Reference)
Cp	Complexity of a program

Created by WE_42

F.Rishan | De Silva D.B.J | Alawathugoda R.M.S.A | Aman M.N.S